**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И МАССОВЫХ КОММУНИКАЦИЙ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ ИМ.**

**ПРОФ. М.А. БОНЧ-БРУЕВИЧА» (СПбГУТ)**

Лабораторная работа №5-6 по курсу

«Логическое и функциональное программирование»

Выполнил: студент группы ИКПИ-14 Хохлов Т.В.

Принял:

доцент кафедры ПИиВТ Ерофеев С.А.

Санкт-Петербург 2023 г.

# Цель работы

Разработать базу данных на языке Turbo Prolog, которая хранит втягивающее для автомобиля Peugeot 408 с двигателем TU5.

# Описание программы

Программа может открывать и сохранять базы данных в формате csv и с помощью файла с предикатами.

Имеются функции редактированния отдельных строк, их добавления и удаления.

Поиск по строкам осуществляется по трем параметрам: по уникальному номеру строки, по совпадению в номере детали или в имени детали.

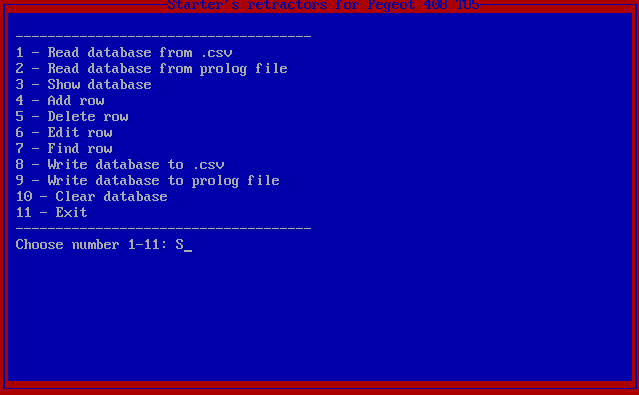
# Используемые предикаты

clear\_database() - отчищает динамическую базу данных choose(integer) — считывает число из консоли и обрабатывает ошибку ввода menu() - предикат выводит пункты меню process(integer) — выполняет опред. Функцию в зависимости от числа search(integer) — выполняет поиск по параматрам в зависимости от числа newLastId(integer) — обновляет глобальную переменную, отвечающую за уникальный индекс новой строки incrementLastId(integer) — прибавляет 1 к глобальной переменной getLastId() - просматривая всю бд, обновляет глобальную переменную readRows() - построчно считывает базу данный из файла csv, и после обработки readValues, заносит новый факт в бд readValue(string,string,string) - разбивают строку на отдельные поля writeCsv() - записывает базу данных в файл csv page(integer) — выводит по 6 строк на экран printRows() - выводит строки printRow(integer) — выводит одну строку printLegend() - печатает названия колонок

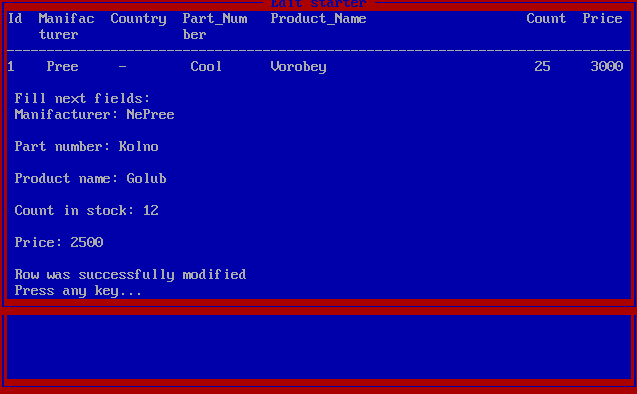
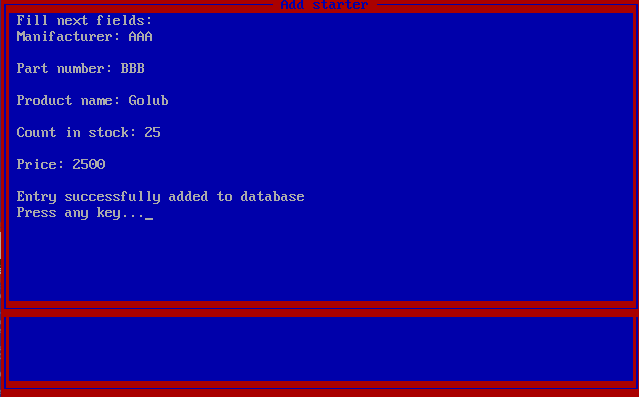
page() - выводит строки пока кол-во строк влезает на экран desicion() - считает кол-во выведенных строк на экран substr(string, string) — проверяет является ли строка подстрокой в строке searchName(string) — выводит строки удовлетворяющие поиску по имени searchPartNumber(string) — выводит строки удовлетворяющие поиску по номеру детали getCountry(string, string) — отображает страну производителя pretty(string,string,string,string,string,string,string) - промежуточный предикат для красивого вывода строки checkLen(integer, string,string,string) — делит строку на до и после введенного инедкса writeSpaces(integer) — выводит введеное кол-во пробелов specialWrite(integer, string) — выводит строку, плюс дополняет пробелами кол-во знаков до введенного числа.

# Тестирование

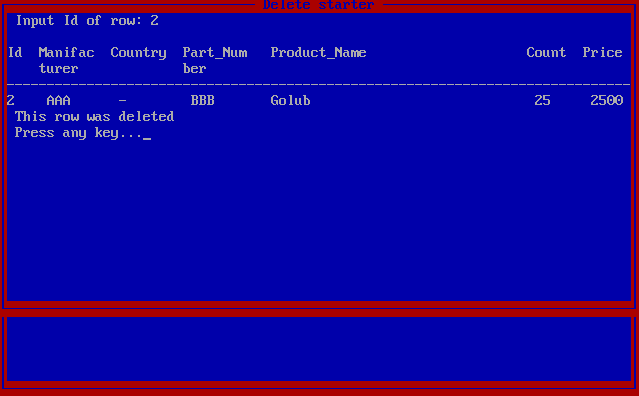
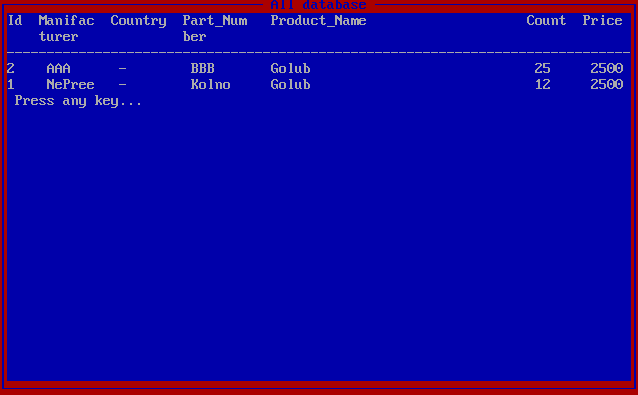
1)Главное меню



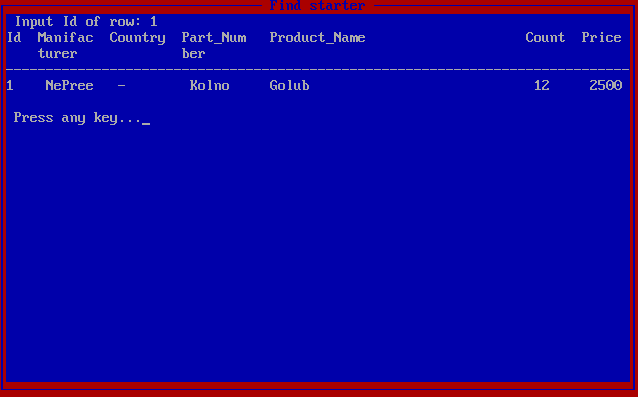
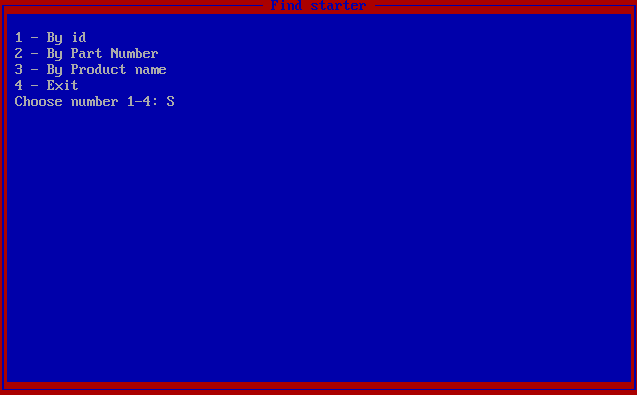
2)Добавление нового вентилятора 3)Изменение строки



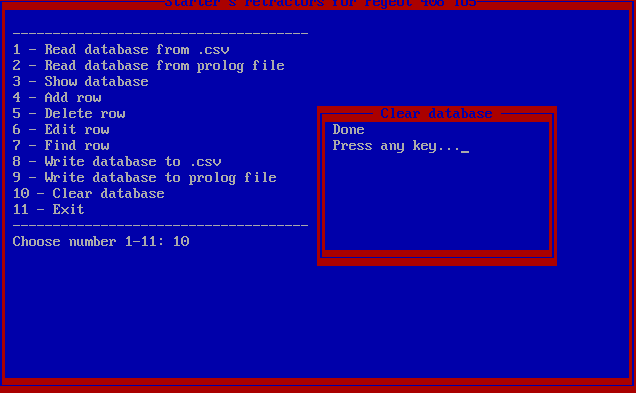
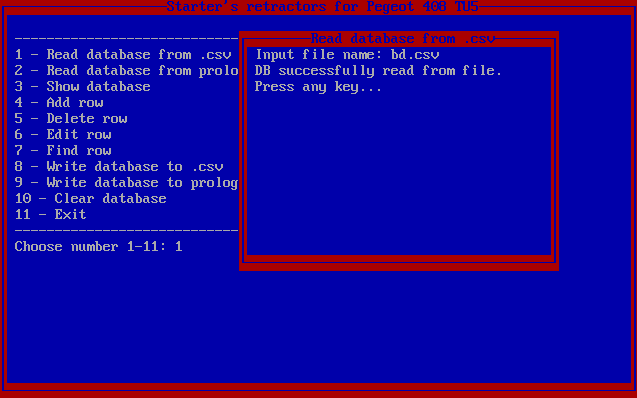
4)Просмотр всей бд 5)Удаление строки



6)Выбор критерия поиска 7)Поиск по уникальному номеру



8)Чтение из csv файла 9)Отчистка базы данных из памяти



# Выводы

В ходе проведенной лабораторной работы я разработал программу на языке Turbo Prolog для хранения базы данных втягивающих на автомобиль Peugeot 408. Было проведено тестирование отдельных предикатов и всей программы в целом. Программа полностью удовлетворяет заданным требованиям.

# Исходный код программы

**domains**

**manifacturer, country, part\_number, name = string**

**id, count = integer**

**price = real**

**file = datafile**

**arr = string\***

**integers = integer\***

**database**

**starter\_retr(id, manifacturer, part\_number, name, count, price)**

**last\_id(integer)**

**rowCount(integer)**

**predicates**

**clear\_database()**

**clear\_vars()**

**choose(integer)**

**menu()**

**process(integer)**

**search(integer)**

**newLastId(integer)**

**incrementLastId(integer)**

**getLastId()**

**readRows()**

**readValue(string,string,string)**

**writeCsv()**

**printRows()**

**printRow(integer, integer)**

**printLegend()**

**page(integer)**

**decision(integer)**

**substr(string, string)**

**substr(string,string,string,string)**

**comparison(string,string,integer)**

**searchName(string)**

**searchPartNumber(string)**

**getCountry(string, string)**

**pretty(string,string,string,string,string,string,string,integer)**

**checkLen(integer, string,string,string)**

**writeSpaces(integer)**

**specialWrite(integer, string)**

**clauses**

**getCountry("Meat&Doria", "Italy").**

**getCountry(\_, "-").**

**substr(\_,""):-!.**

**substr("",\_):-!,fail.**

**substr(S,SS):-**

**frontchar(S,SC,S\_),**

**frontchar(SS,SC,SS\_),**

**substr(S\_,SS\_),!.**

**substr(S,SS):-**

**frontchar(S,\_,S\_),**

**substr(S\_,SS),!.**

**substr(\_,"",R,R):-!.**

**substr("",\_,\_,""):-!.**

**substr(S,SS,RT,R):-**

**frontchar(S,SC,S\_),**

**frontchar(SS,SC,SS\_),**

**substr(S\_,SS\_,RT,R),!.**

**substr(S,SS,\_,R):-**

**frontchar(S,\_,S\_),**

**substr(S\_,SS,S\_,R),!.**

**comparison("","",0):-!.**

**comparison(\_,"",1):-!.**

**comparison("",\_,-1):-!.**

**comparison(S1,S2,R):-**

**frontchar(S1,C,S1\_),**

**frontchar(S2,C,S2\_),**

**comparison(S1\_,S2\_,R);**

**frontchar(S1,C1,\_),**

**frontchar(S2,C2,\_),**

**C1<C2,R=-1,!;**

**R=1,!.**

**choose(X) :-**

**readint(X);**

**write("Wrong number"), readchar(\_), X = 0,**

**removewindow, menu.**

**clear\_vars() :-**

**retract(rowCount(\_)),**

**retract(last\_id(\_)),**

**fail.**

**clear\_vars():-!.**

**clear\_database:-**

**retract(starter\_retr(\_,\_,\_,\_,\_,\_)),**

**fail.**

**clear\_database:-clear\_vars(),!.**

**incrementLastId(X) :-**

**retract(last\_id(Id)),**

**X = Id + 1,**

**assert(last\_id(X));**

**X = 1,**

**assert(last\_id(X)).**

**newLastId(X) :-**

**last\_id(Last), Last < X,**

**retract(last\_id(\_)),**

**assert(last\_id(X)),!.**

**newLastId(X) :-**

**assert(last\_id(X)).**

**decision(Height) :-**

**not(rowCount(\_)), assert(rowCount(5)),fail;**

**rowCount(Cnt),Height + Cnt > 22,**

**retract(rowCount(Cnt)), assert(rowCount(0));**

**rowCount(Cnt), retract(rowCount(Cnt)),**

**NewCnt = Height + Cnt, assert(rowCount(NewCnt)),fail,!.**

**page(Height) :-**

**decision(Height),**

**readchar(A), clearwindow,**

**A = 27.**

**printRows() :-**

**starter\_retr(X1,\_,\_,\_,\_,\_),**

**printRow(X1,Height),**

**page(Height).**

**printRows().**

**printLegend() :-**

**write("Id Manifac Country Part\_Num Product\_Name Count Price"),nl,**

**write(" turer ber"),nl,**

**write("------------------------------------------------------------------------------").**

**printRow(Id, Height) :-**

**starter\_retr(Id, X2, X3, X4, X5, X6),**

**str\_int(IdString, Id), str\_int(CountString, X5), str\_real(PriceString, X6),**

**getCountry(X2, Country),**

**pretty(IdString, X2, Country, X3, X4, CountString, PriceString, Height),!.**

**specialWrite(Len, String) :-**

**str\_len(String, StrLen),**

**StrLen = Len, write(String),!;**

**str\_len(String, StrLen),**

**SUB = Len - StrLen,**

**write(String), writeSpaces(SUB).**

**writeSpaces(0).**

**writeSpaces(X) :-**

**New = X - 1, write(" "),**

**writeSpaces(New).**

**checkLen(Len, String, A, B) :-**

**str\_len(String, StrLen),**

**StrLen <= Len,**

**A = String, B = "",!;**

**frontstr(Len, String, A, B),!.**

**pretty("","","","","","","", Height) :- Height = 0.**

**pretty(X1,X2,X3,X4,X5,X6,X7,Height) :-**

**checkLen(3, X1, X11, X12), specialWrite(3, X11), write(" "),**

**checkLen(7, X2, X21, X22), specialWrite(7, X21), write(" "),**

**checkLen(7, X3, X31, X32), specialWrite(7, X31), write(" "),**

**checkLen(8, X4, X41, X42), specialWrite(8, X41), write(" "),**

**checkLen(31,X5, X51, X52), specialWrite(31,X51), write(" "),**

**checkLen(5, X6, X61, X62), specialWrite(5, X61), write(" "),**

**checkLen(5, X7, X71, X72), specialWrite(5, X71),**

**pretty(X12,X22,X32,X42,X52,X62,X72, New),**

**Height = New + 1.**

**writeCsv() :-**

**starter\_retr(X1,X2,X3,X4,X5,X6),**

**writef("%d;%s;%s;%s;%d;%f\n",X1,X2,X3,X4,X5,X6), fail.**

**writeCsv().**

**readRows() :-**

**readln(Line),**

**readValue(Line, X1, Tail1), str\_int(X1, Id),**

**readValue(Tail1, X2, Tail2),**

**readValue(Tail2, X3, Tail3),**

**readValue(Tail3, X4, Tail4),**

**readValue(Tail4, X5, Tail5), str\_int(X5, Count),**

**readValue(Tail5, X6, \_), str\_real(X6, Price),**

**assertz(starter\_retr(Id, X2, X3, X4, Count, Price)), newLastId(Id), !,**

**readRows();**

**not(eof(datafile)), !,**

**write(" Reading error"),nl,**

**write(" Only some data opened"),nl;**

**!.**

**getLastId() :-**

**not(last\_id(\_)), assert(last\_id(0)),fail.**

**getLastId() :-**

**starter\_retr(Id,\_,\_,\_,\_,\_),**

**newLastId(Id),fail.**

**getLastId() :- !.**

**readValue("", "", "") :- !.**

**readValue(Line, X, Tail) :-**

**frontchar(Line, LineH, LineT),**

**LineH = ';', !,**

**X = "", Tail = LineT;**

**frontchar(Line, LineH, LineT),**

**LineH <> ';',**

**readValue(LineT, T, Tail),**

**str\_char(LineHS, LineH),**

**concat(LineHS, T, X).**

**menu :-**

**makewindow(1,23,65, "Starter's retractors for Pegeot 408 TU5",0, 0,25,80),**

**clearwindow,nl,**

**write(" ------------------------------------- "),nl,**

**write(" 1 - Read database from .csv"), nl,**

**write(" 2 - Read database from prolog file"), nl,**

**write(" 3 - Show database"), nl,**

**write(" 4 - Add row "), nl,**

**write(" 5 - Delete row "), nl,**

**write(" 6 - Edit row"), nl,**

**write(" 7 - Find row"), nl,**

**write(" 8 - Write database to .csv"), nl,**

**write(" 9 - Write database to prolog file"), nl,**

**write(" 10 - Clear database"), nl,**

**write(" 11 - Exit"), nl,**

**write(" ------------------------------------- "),nl,**

**write(" Choose number 1-11: "),**

**choose(X), nl, process(X).**

**process(1) :-**

**makewindow(2,23,65, "Read database from .csv", 2, 30, 15, 40),**

**write(" Input file name: "), readln(FileName),**

**concat("C:/BDS/", FileName, File),**

**existfile(File), clear\_database,**

**openread(datafile, File), readdevice(datafile),**

**readRows(),**

**closefile(datafile), readdevice(keyboard),**

**write(" DB successfully read from file."), nl, !,**

**write(" Press any key..."), readchar(\_),**

**removewindow, clearwindow, menu;**

**write(" Error reading file!"), nl, !,**

**write(" Press any key..."), readchar(\_),**

**removewindow, clearwindow, menu, fail.**

**process(2) :-**

**makewindow(3,23,65, "Read database from prolog file", 2, 30, 15, 40),**

**write(" Input file name: "), readln(FileName),**

**concat("C:/BDS/", FileName, File),**

**existfile(File), clear\_database,**

**consult(File),**

**getLastId(),**

**write(" DB successfully read from file."), nl, !,**

**write(" Press any key..."), readchar(\_),**

**removewindow, clearwindow, menu;**

**write(" Error reading file!"), nl, !,**

**write(" Press any key..."), readchar(\_),**

**removewindow, clearwindow, menu, fail.**

**process(3) :-**

**makewindow(4,23,65, " All database ",0, 0,25,80), shiftwindow(4),**

**printLegend(), printRows(),**

**write(" Press any key..."), readchar(\_),**

**removewindow, clearwindow, menu;**

**write(" Press any key..."), readchar(\_),**

**removewindow, clearwindow, menu.**

**process(4) :-**

**makewindow(5,23,65, " Add starter " ,0,0, 20,80), shiftwindow(5),**

**write(" Fill next fields:"),nl,**

**write(" Manifacturer: "), readln(Manifacturer),nl,**

**write(" Part number: "), readln(PartNumber),nl,**

**write(" Product name: "), readln(Name),nl,**

**write(" Count in stock: "), readint(Count), Count >= 0, nl,**

**write(" Price: "), readreal(Price), Price >= 0, nl,**

**incrementLastId(Id),**

**assertz(starter\_retr(Id, Manifacturer, PartNumber, Name, Count, Price)),**

**write(" Entry successfully added to database"),nl, !,**

**write(" Press any key..."), readchar(\_),**

**removewindow, clearwindow, menu;**

**write(" Something went wrong"),nl,**

**write(" Press any key..."), readchar(\_),**

**removewindow, clearwindow, menu, fail.**

**process(5) :-**

**makewindow(6,23,65, " Delete starter " ,0,0,20,80), shiftwindow(6),**

**write(" Input Id of row: "), readint(Id),nl,**

**printLegend(), printRow(Id,\_),**

**retract(starter\_retr(Id,\_,\_,\_,\_,\_)),**

**write(" This row was deleted"),nl, !,**

**write(" Press any key..."), readchar(\_),**

**removewindow, clearwindow, menu;**

**write(" No data"), nl, !,**

**write(" Press any key..."), readchar(\_),**

**removewindow, clearwindow, menu, fail.**

**process(6) :-**

**makewindow(7,23,65, " Edit starter " ,0,0,20,80), shiftwindow(7),**

**write(" Input Id of row: "), readint(Id),**

**printLegend(), printRow(Id,\_), nl,**

**write(" Fill next fields:"),nl,**

**write(" Manifacturer: "), readln(Manifacturer),nl,**

**write(" Part number: "), readln(PartNumber),nl,**

**write(" Product name: "), readln(Name),nl,**

**write(" Count in stock: "), readint(Count), Count >= 0, nl,**

**write(" Price: "), readreal(Price), Price >= 0, nl,**

**retract(starter\_retr(Id,\_,\_,\_,\_,\_)),**

**assertz(starter\_retr(Id, Manifacturer, PartNumber, Name, Count, Price)),**

**write(" Row was successfully modified"),nl, !,**

**write(" Press any key..."), readchar(\_),**

**removewindow, clearwindow, menu;**

**write(" Wrong input"), nl, !,**

**write(" Press any key..."), readchar(\_),**

**removewindow, clearwindow, menu, fail.**

**process(7) :-**

**makewindow(8,23,65, " Find starter ",0, 0,25,80), shiftwindow(8),**

**clearwindow,nl,**

**write( " 1 - By id"), nl,**

**write( " 2 - By Part Number"), nl,**

**write( " 3 - By Product name"), nl,**

**write( " 4 - Exit"), nl,**

**write(" Choose number 1-4: "),**

**choose(X), clearwindow,**

**search(X),!,**

**write(" Press any key..."), readchar(\_),**

**removewindow, clearwindow, menu;**

**write(" Wrong input"), nl, !,**

**write(" Press any key..."), readchar(\_),**

**removewindow, clearwindow, menu, fail.**

**process(8) :-**

**makewindow(9,23,65, "Write database to .csv", 2, 30, 15, 40), shiftwindow(9),**

**write("Input file name: "), readln(FileName),**

**concat("C:/BDS/", FileName, File),**

**openwrite(datafile, File), writedevice(datafile),**

**writeCsv(),**

**closefile(datafile), writedevice(screen),**

**write(" DB successfully saved to file."), nl, !,**

**write(" Press any key..."), readchar(\_),**

**removewindow, clearwindow, menu;**

**write(" Error writing database!"), nl, !,**

**write(" Press any key..."), readchar(\_),**

**removewindow, clearwindow, menu, fail.**

**process(9) :-**

**makewindow(10,23,65, "Write database to prolog file", 2, 30, 15, 40), shiftwindow(10),**

**write("Input file name: "), readln(FileName),**

**concat("C:/BDS/", FileName, File),**

**save(File),**

**write(" DB successfully saved to file."), nl, !,**

**write(" Press any key..."), readchar(\_),**

**removewindow, clearwindow, menu;**

**write(" Error writing database!"), nl, !,**

**write(" Press any key..."), readchar(\_),**

**removewindow, clearwindow, menu, fail.**

**process(10) :-**

**makewindow(11,23,65, " Clear database " ,7,40,10,30), shiftwindow(11),**

**clear\_database,**

**write(" Done "), nl,**

**write(" Press any key..."), readchar(\_),**

**removewindow, clearwindow, menu;**

**write(" Error while delete db "), nl,**

**write(" Press any key..."), readchar(\_),**

**removewindow, clearwindow, menu, fail.**

**process(11) :-**

**write("See you again! "),readchar(\_),exit.**

**search(1) :-**

**write(" Input Id of row: "), readint(Id),**

**printLegend(), printRow(Id,\_),nl,!;**

**write(" No entries found"),nl,!,**

**write(" Press any key..."), readchar(\_),**

**removewindow, clearwindow, process(7), fail.**

**search(2) :-**

**write(" Input part number: "), readln(Line),**

**clearwindow,write(" Press any key to see next result or ESC to finish..."), nl,**

**printLegend(),**

**searchPartNumber(Line),!;**

**write(" No entries found"),nl,**

**write(" Press any key..."), readchar(\_),**

**removewindow, clearwindow, process(7), fail.**

**search(3) :-**

**write(" Input product name: "), readln(Line), nl,**

**clearwindow,write(" Press any key to see next result or ESC to finish..."), nl,**

**printLegend(),**

**searchName(Line),!;**

**write(" No entries found"),nl,**

**write(" Press any key..."), readchar(\_),**

**removewindow, clearwindow, process(7), fail.**

**search(4) :-**

**removewindow, clearwindow, menu.**

**searchName(Line) :-**

**starter\_retr(Id,\_,\_,Name,\_,\_), substr(Name, Line),**

**printRow(Id,\_),**

**readchar(A), A = 27,**

**removewindow, clearwindow, menu.**

**searchPartNumber(Line) :-**

**starter\_retr(Id,\_,PartNumber,\_,\_,\_), substr(PartNumber, Line),**

**printRow(Id,\_),**

**readchar(A), A = 27,**

**removewindow, clearwindow, menu.**

**goal**

**menu.**